**FORMAN CHRISTIAN COLLEGE (A CHARTERED UNIVERSITY)**

****

**COMP 451 (Compiler Construction)**

**2022 FALL**

**Lab - 6**

**Gulraiz Noor Bari (231-525536)**

**Muhammad Hamza (231-450349)**

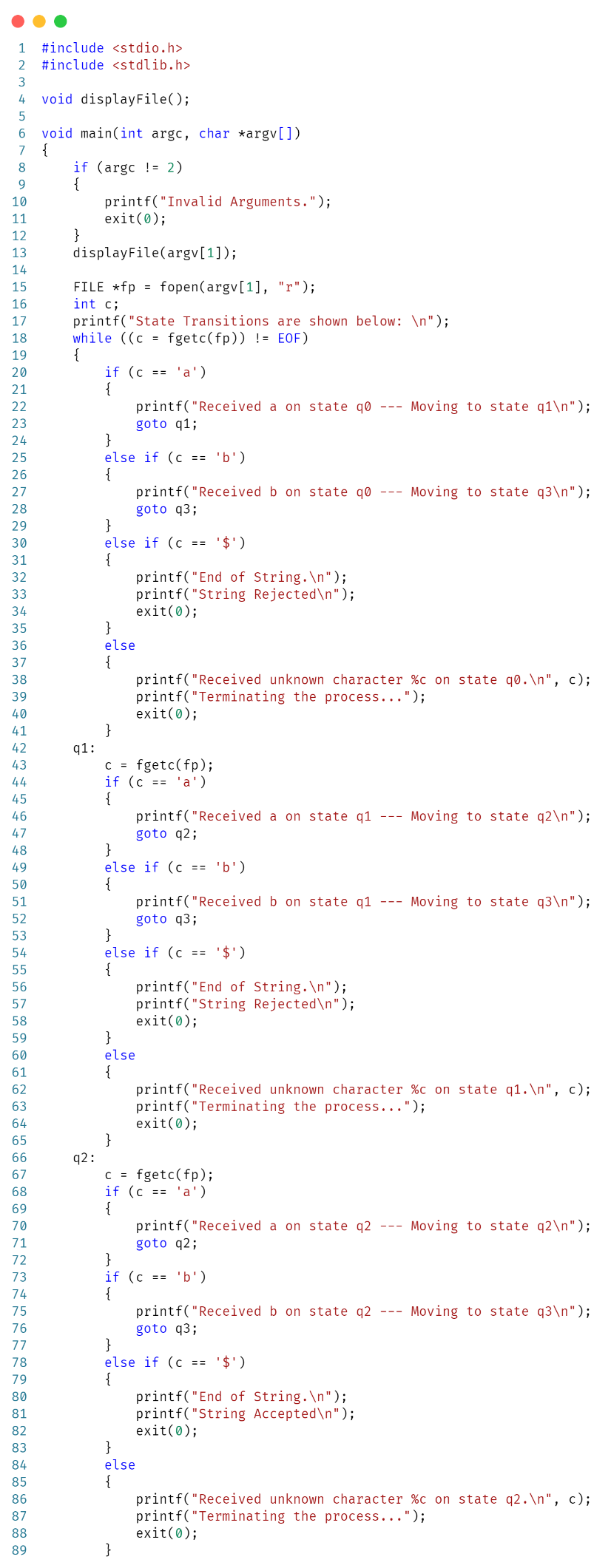
**INTRODUCTION:**

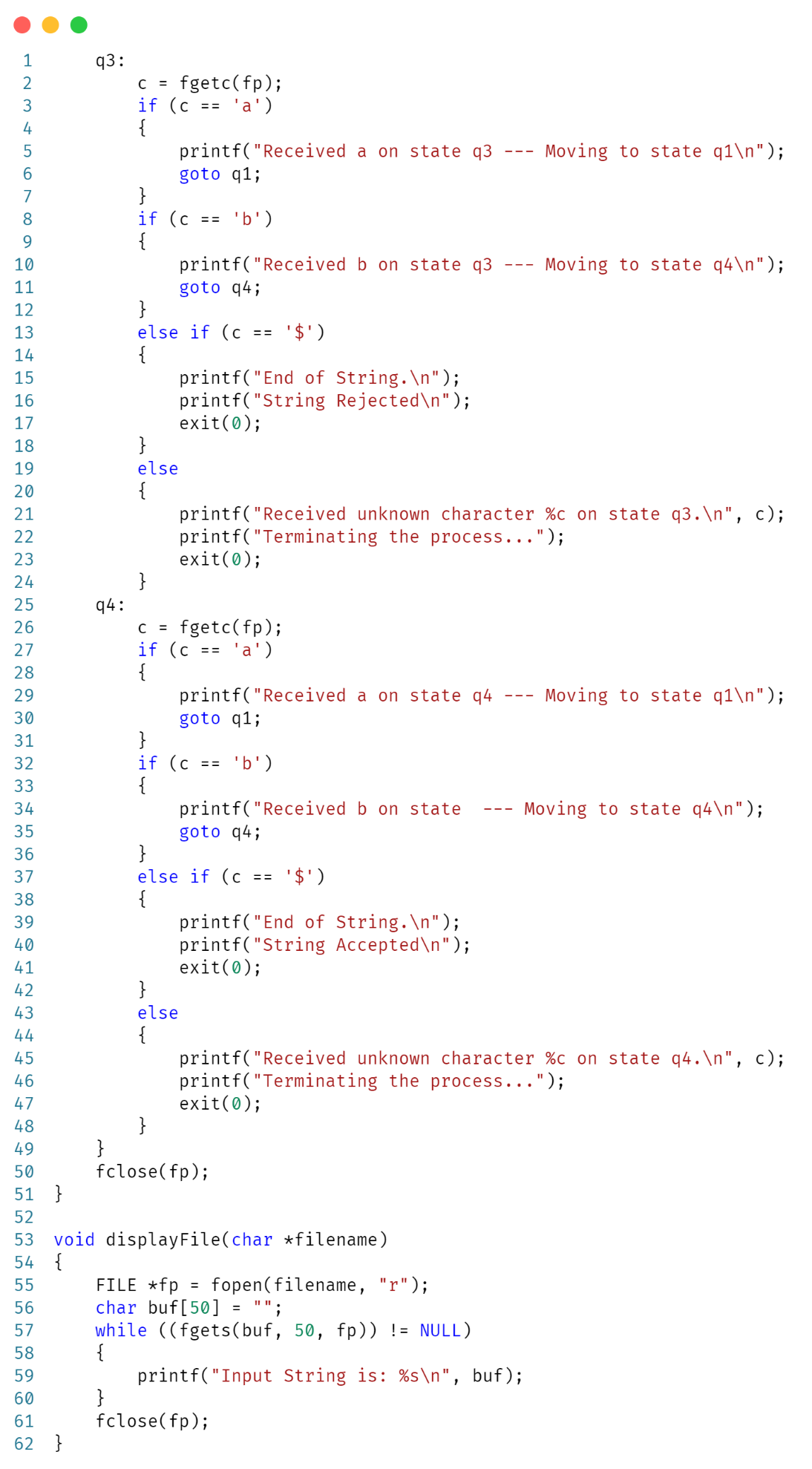
* **stdio.h** (standard input/output)is a header file that contains declarations for functions like, printf, scanf, etc.
* **stdlib.h** (standard library) is a header file that contains declarations of functions that involves memory allocation and process control. For example, in our program we used **exit(0)** at the end; which means to successfully terminate the program.
* **main():** The main function serves as the starting point of the program execution in C language. User can pass any number of parameters depending upon the requirements of the program logic or structure. We have passed two parameters:
* **int argc:** Keeps count of the number of command-line arguments entered by the user that also includes the name of the program.
* **char \* argv[]:** Here argv (stores command-line arguments) is an array of pointers that is pointing to the characters of the command-line arguments entered by the user.

**LOGIC/ALGORITHM:**

The code is a program that reads a file specified by the user that contains a string and validates using DFA. The DFA accepts any string that ends with either **‘aa’** or **‘bb’** and rejects all other types of strings.

* The program starts by checking whether two arguments were passed by the user in command line, which is done by argc (which has the count of arguments entered in the command-line). If the number of arguments is not two the function is not proceeded and displays a message “Invalid argument”, otherwise if user has entered two arguments; in our case the executable and file name that is to be read, the program proceeds.
* First the display function is called which takes a file pointer name as input and reads and displays the contents of the file, line-by-line using **fgets()**.
* Then in the **main(),** the file is opened using **fopen** function which takes two parameters; filename and the mode in which the file should be opened, it returns a pointer pointing to the address in the memory where the file is stored. In our case “fp” holds that file pointer that points to the file stored in argv[1], as the file to read is the second argument user enters in the command-line.
* The file is read by character-by-character and for each character following operations are performed:
  + If the character being read is either **‘a’** or **‘b’** then we move on the next specified state using **goto statement** (jumps directly to the specified statement).
  + If the character being read is **‘$’** it terminates the string and either rejects or accepts the string depending whether the string being read is on the final or non-final state of the DFA.
  + If the character is not defined in the state transition table, then it is considered as an unknown character, hence program handles this by throwing an error statement and terminating the program.
* After jumping to other state e.g; q0 to q1, we repeat the above checks, the program will terminate on the following conditions:
  + Whenever the character being read becomes equal to sentinel symbol **‘$’.** String is accepted if it ends on states q2 and q4 (as they are the final states) and string is rejected if it ends on states q1 and q3 (as they are the non-final states of the DFA).
  + If unknown character is encountered.
* At the end of the loop program prints the transitions that were made and the status of string (accepted or rejected).
* The file is closed using **fclose()** and program ends with a **exit(0)** statement, which refers to successful termination of the program.

****



**OUTPUT:**

